INTEGRATION OF SIMULATION MODELING AND COMPUTER AIDED PRODUCTION MANAGEMENT IN COMPUTER INTEGRATED ENTERPRISE

Emine Persentili and Sema Alptekin

Engineering Management Department
University of Missouri - Rolla

ABSTRACT

Designing efficient and integrated manufacturing systems is the first step in attaining computer integrated enterprises (CIE). Integration of planning and implementation phases of manufacturing is essential for taking full advantage of the CIE. In order to design reliable and efficient manufacturing systems, the designers must consider the impacts of planning decisions made by computer aided production management (CAPM) modules on implementations held in manufacturing cells.

This paper focuses on two issues. The first issue is importance and necessity of integrating CAPM modules with manufacturing cells. The second issue includes major features of the object-oriented approach and their relevance to our objective of modeling a design framework which focuses on integration of CAPM modules and simulation models which emulate the manufacturing cells in the CIE environment.

INTRODUCTION

Developing a computer integrated enterprise (CIE) involves the automation and integration of all elements of the manufacturing system by means of communication networks and databases. All manufacturing operations, and such computer-based tools as hardware and software utilized in these operations, are the elements of manufacturing systems. For CIE to be effective, each element must be automated in itself first, and then integrated with other elements in a way that its generated outputs will be the required inputs of the others. Therefore, designing efficient CIEs, with all their diverse hardware and software components, is very complex.

As argued by Adiga [1], getting the most out of CIE depends on several factors, including the development of suitable software modeling approaches to support efficient design and control of manufacturing systems. In this area, much concern has been expressed for the development of computer aided design (CAD), factory automation, and process planning elements of CIE. However, there are still several problems in planning and controlling functions held by the CAPM element of CIE. Some of these problems are those of sequencing and scheduling of operations in manufacturing cells, capacity selection, resource (machine and equipment) allocation, and batch sizing. In order to design complete CIE systems, more attention must be paid to the integration of CAPM functions, without which making full use of all islands of automation and computer aided management techniques in an enterprise is unrealistic [2].

CAPM, in its simplest form, deals with balancing schedule and capacity by taking into consideration management policies of the enterprise. The balancing act is presently performed at two levels: first at the planning level and second at the implementation level. The gap between these two levels causes problems of shortages, overages, production delays, capacity imbalances, low machine utilization, and significant efficiency losses in manufacturing cells. In order to narrow the gap between these two levels, we propose to integrate production management functions with manufacturing processes during both the design and operation phases of an enterprise. This integration would enable the enterprise to achieve cost-efficient and effective manufacturing systems resulting in timely production and distribution of products.

The object-oriented philosophy is a promising approach for modeling and designing large systems such as integrated manufacturing systems. It is faster to build, and easier to maintain and extend object oriented systems. Object oriented systems operate ideally in a distributed enterprise network environment. They reduce the massive blocks of code by linking data and procedures. Thus, we propose using an object-oriented approach to design and operate CIEs that integrate major CAPM functions with manufacturing cell operations.

In this paper, we first discuss the operations of main CAPM modules and their interactions with manufacturing cells. Then, the main features of the object-oriented approach are presented, with their matching points with manufacturing system design. In the last section, the scope of our research is summarized.

INTERACTIONS BETWEEN CAPM AND MANUFACTURING CELLS

CAPM manages the time- and capacity-related planning tasks in such a way that resources are assigned to the operations in the manufacturing cell in order to meet some relevant criteria, such as when and how much to produce for meeting due dates. CAPM consists of several modules. The main modules, their functions, and their interactions with each other and with manufacturing cells are discussed below.

Master production schedule (MPS):

This module generates a description of what is to be manufactured, when it is to be delivered and in what

quantity, based on order information, product planning and design information, and overall business objectives. The MPS must provide final outputs to the material requirement planning and the capacity requirement planning modules.

Material requirement planning (MRP):

This module receives the base information from MPS for calculating future requirements of material at every stage of manufacture. The goal of MRP is to tie the movement of materials needed to manufacture products with the requirement to build such products and to ensure that material will be available to meet the needs of the manufacturing cells. Functionally, an MRP system produces a time-phased picture of inventory movement and, in doing so, provides the input needed to generate the orders for replenishment of inventory.

Capacity requirement planning (CRP):

This module receives all the suggested and existing orders, needed to make the products and their components, from MRP. The CRP has a broad responsibility to provide that capacity at the time needed and to prevalidate schedules build in MPS and MRP, which assume a particular capacity.

All these activities of organizing, planning, coordinating, and controlling the production operations builds a wide transaction relationship between CAPM and manufacturing cells. The output of CAPM modules is broken down to shop orders for each resource (machine and equipment) in manufacturing cells. This is the information input of manufacturing cells for preparing detailed schedules of resources to optimize efficiency and machine utilization. Status information of the manufacturing cells is the feedback data for the renewal of the CAPM decisions. Integration of all this information will result in significant improvements in the overall performance of CIEs. As summarized in the following section, object-oriented modeling offers promising advantages in achieving this integration.

OBJECT ORIENTED MODELING AND DESIGN

The object-oriented approach views the world as a collection of discrete objects which act and react in a common environment by exchanging messages. Object-oriented programming (OOP) involves designing software around these objects by giving data a more central position in program organization, and tying the data more closely to the functions that act on it. OOP is characterized by many concepts: encapsulation, data hiding, classification, inheritance, data abstraction and polymorphism. These concepts, together with object definitions, are basic to the development of software that satisfy the requirements of advanced manufacturing systems, such as; high level of software reusability, software modularity and the ability to implement large, detailed models in an understandable way [1,3-5]. Brief definitions of major OOP concepts and their advantages in designing manufacturing systems are presented hereafter.

Object:

The object is a collection of some discrete data and a set of operations that can access that data. Objects may correspond to conceptual or physical entities of interest in the system. For example, objects can be thought of as representing machines that are capable of performing some predefined actions in response to requests. A message is a request for an object to carry out one of its operations. There is a set of operations (methods) that the objects know how to perform. Objects store data in variables and respond to messages by executing methods. In the object-oriented approach, the data and procedures directly affecting an object are bundled to enable users to have a natural one-to-one correspondence between software objects and such manufacturing objects as machines, equipment, lots, and parts. This helps to make more accurate representations of the real manufacturing environment in an object-oriented programming system. The use of objects and messages allows achievement of easily understandable system designs.

Encapsulation and data hiding:

Encapsulation is the process of binding together the data with the functions that operate on it. Data hiding is ensuring that the data in an object cannot be accessed by other objects and only be accessible through a prescribed message interface. The details of how data is stored and what algorithm is used to return a reply to a message is also hidden.

A program developer needs to understand the object interface or the sets of methods available for that object class. The developer also needs to know what each method requires in order to operate and what state the object is in after the operation is performed. It is important that the documentation for any object interface contain this information in a very accessible way. Encapsulation within software objects of local data, representing the state of the physical object and methods for updating the state variables, provides modularity of software.

Classification and inheritance:

Every object in the OOP has a specific type that is classified in a hierarchy. Where an object belongs in the hierarchy depends on the messages it responds to, and its descriptive attributes. The creation of a new class as an extension or specialization of an existing class is referred to as inheritance. It allows the conceptual relationship between different classes to be made explicit. Inheritance allows easy comprehension of relationships between classes and also much easier construction of the classes themselves. Classification and inheritance features enhance reusability of code, since existing code need not be modified at all. Existing classes can simply inherit the changes made to the derived class. Thus, these features make it possible to change a running system incrementally and still ensure that it works. The objects in the new class will recognize all the messages of the original parent class as well as new messages to implement new functions, so that other objects that work with the original simple versions will still work with the new objects.

It is possible also to build prototype systems that can be evolved into manufacturing systems without starting from scratch. This makes existing systems easy to modify and extend, by adding new objects, and makes objects created before for plant A, usable for plant B by adding some more objects. There is no need to rewrite the code for the existing objects, so that there is a time saving in developing new applications. Easy modification and extension of OOP is a significant advantage in developing software for manufacturing systems, since the nature of manufacturing systems is dynamically changing as a result of continuous and fast technology developments.

Modularity of objects and the concept of inheritance in the object-oriented approach makes the incremental approach possible in designing manufacturing systems. The incremental approach is very important in designing advanced manufacturing systems which are characterized by complexity and need for detail. The sharing of common

classes and the ability to reuse the classes in different modules will encourage better integration of software modules.

Abstraction:

Inheritance and encapsulation together provide abstraction. Abstraction is essential for dealing with complexity. Very complex systems can be simplified by dealing with them at the appropriate level of detail. Thus, a system can be designed such that a user may have the option of running the overall program of a factory, a department or a single manufacturing cell. Abstraction on multiple levels is required in modeling manufacturing systems, because people at different levels in the organizational hierarchy tend to view data at different levels of abstraction.

Polymorphism:

This concept is equivalent to saying that each object is a specific example of some generic object. It allows us to send identical messages to different objects and have each object respond in ways appropriate to their individual behaviors. Thus, different objects can share common interfaces.

All these features make the object-oriented approach more advantageous than conventional programming in developing effective computer programs for dynamically changing, complex manufacturing systems, containing a large amount of detailed interactions. Our research utilizes the object-oriented approach in designing manufacturing systems that focus on integrating CAPM modules with each other and with the manufacturing cell. In the next section, the scope of our research is presented.

SYNOPSIS OF THE ONGOING RESEARCH

The main purpose of our research is to develop a framework in modeling the following two systems and the interactions between them by utilizing the object-oriented approach: (1) the functions held in MPS, MRP, and CRP modules, (2) the simulation of manufacturing cells.

Applying OOP to model these systems involves identification of objects in the system, their operations, and their relations with other objects. Objects are the nuclei of our framework. An object may be anything that plays a specific role of interest in the system. Objects of interest are those that are capable of performing some predefined operations or actions when requested by a message from other objects. In our framework, the object classes in the highest level of the class hierarchy are MPS, MRP, CRP modules of CAPM, and simulation models of manufacturing cells (see Figure 1). Whenever any object requires an action or response from another object, it is achieved by sending a message to the appropriate object. Each CAPM object is characterized by a sequence of messages representing their functions.

One of the features of this framework is the ability to emulate the CIE information flow logic to perform what-if and try-for-fit analyses on objects, based on the modifications made in other objects. These objects will be integrated in the network environment as it can be seen in Figure 2, based on messages linking their data and methods. Thus, modifications made in data and methods of one object will access the network immediately and other objects will be updated automatically based on these modifications. For example, possible modifications in the objects are the changes in customer satisfaction policy in MPS, the changes in lot sizing and inventory policies in MRP, and the changes in machine and capacity selection decisions in CRP. Each modification impacts the other modules based on interactions among them. As a response to the decisions held in CAPM, such results of implementations as scheduling and machine utilization in manufacturing cells are feedback to re-evaluate CAPM decisions.

This framework will be implemented in a prototype model, which is the automated and integrated manufacturing cell in the Computer Integrated Manufacturing (CIM) laboratory of the Engineering Management Department-University of Missouri-Rolla. The component machines of this manufacturing cell are a GMF robot, an IBM robot, a Bridgeport CNC milling machine, an automated storage and retrieval system (AS/RS), and a loop conveyor, all integrated and controlled by programmable controllers and microcomputers. This manufacturing cell is capable of producing several items, such as an ice scraper, CD rack, key chain, and desk organizer.

Object classes are described in the class hierarchy of the object-oriented system as it can be seen in Figure 1. Object classes defined for MPS, MRP, CRP, and manufacturing cell may be further subclassed. For example, the manufacturing cell is decomposed into machine, product, and simulation processing object classes. Machine and product are also decomposed into the object classes which correspond to the machines utilized, and products produced in the manufacturing cell. Describing these objects as classes will make it possible to form other objects as variations of these classes in the future. These classes have a hierarchical structure and each class inherits methods from upper-level classes. This approach will also allow us to address interactions between objects at different levels of abstraction, as is demonstrated in Figure 1.

REFERENCES

[1] Adiga S., Software Modeling of Manufacturing Systems: A Case for an Object-Oriented Programming Approach, Annals of Operations Research, 1989, pp. 363-378.

[2] Zhou H., The Role of Computer-Aided ProductionManagement in CIM systems, Computers in Industry, 1992, 19, pp. 119-126.

[3] Adiga S., Lin W., Object Oriented Simulation of Manufacturing Systems, 1989 IIE Integrated Systems Conference and Society for Systems, Integrated Manufacturing Conference Proceedings, pp. 489-494.

[4] Adiga S., Glassey C. R., Object Oriented Simulation to Support Research in Manufacturing Systems, International Journal of Production Research, 1991, 29, 12, pp. 2529-2542.

[5[Rumbaugh J., Michael B., Bremerlani W., Eddy F., Lorensen W., Object-Oriented Modeling and Design, Prentice Hall, 1991.

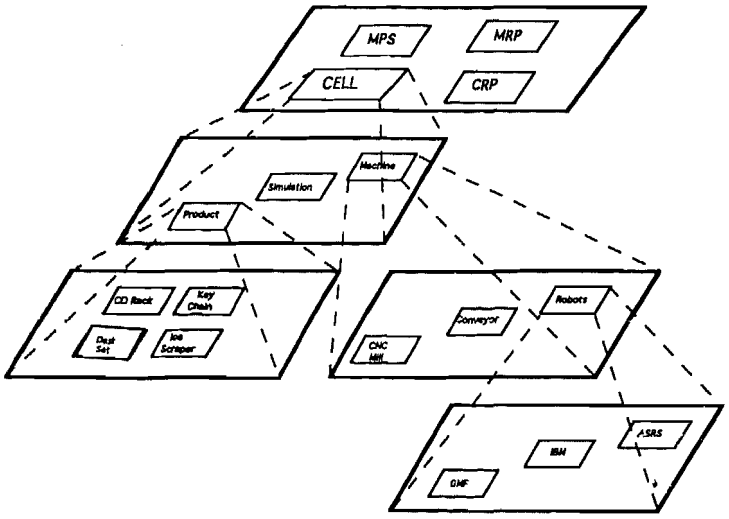

Figure 1. Representation of Object Class Hierarchy

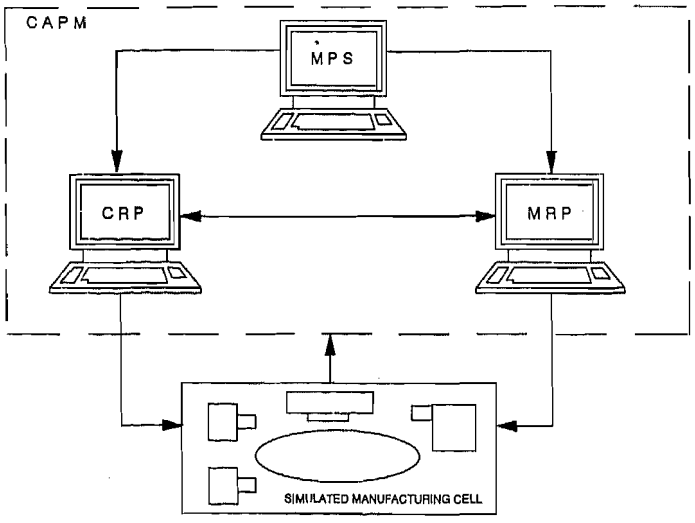

Figure 2. Integration of Object Classes

This work is supported in part by the NSF Grant #USE-9054210.